

Exploiting PUF Models for Error Free Response Generation

Yansong Gao^{†‡}, Hua Ma[†], Gefei Li[†], Shaza Zeitouni[§], Said F. Al-Sarawi[‡], Derek Abbott[‡], Ahmad-Reza Sadeghi[§] and Damith C. Ranasinghe[†]

[‡]School of Electrical and Electronic Engineering, The University of Adelaide, SA 5005, Australia
{yansong.gao, said.alsarawi, derek.abbott}@adelaide.edu.au

[†]Auto-ID Labs, School of Computer Science, The University of Adelaide, SA 5005, Australia
{mary.ma, gefei.li, damith.ranasinghe}@adelaide.edu.au

[§]System Security Lab, Technische Universitat Darmstadt, Darmstadt 64289, Germany
shaza.zeitouni@trust.cased.de; ahmad.sadeghi@trust.tu-darmstadt.de

Abstract—Physical unclonable functions (PUF) extract secrets from randomness inherent in manufacturing processes. PUFs are utilized for basic cryptographic tasks such as authentication and key generation, and more recently, to realize key exchange and bit commitment requiring a large number of error free responses from a strong PUF. We propose an approach to eliminate the need to implement expensive on-chip error correction logic implementation and the associated helper data storage to reconcile naturally noisy PUF responses. In particular, we exploit a statistical model of an Arbiter PUF (APUF) constructed under the nominal operating condition during the challenge response enrollment phase by a trusted party to judiciously select challenges that yield error-free responses even across a wide operating conditions, specifically, a $\pm 20\%$ supply voltage variation and a 40°C temperature variation. We validate our approach using measurements from two APUF datasets. Experimental results indicate that large number of error-free responses can be generated on demand under worst-case when PUF response error rate is up to 16.68%.

Index Terms—Error free response, PUF, Modeling building

I. INTRODUCTION

Physical unclonable functions (PUFs) are low-cost hardware security primitives and can be seamlessly integrated with devices during the design and fabrication processes [1], [2], [3], to act as trust anchors. The PUF, in essence, extracts secrets from the inevitable process variations. Hence, in reality, identical PUF instances cannot be forged, not even by the same manufacturer. There exists a number of PUF structures [4], [5], [6], [7]. Among them, time-delay based PUFs [5], [1], in particular, the Arbiter PUF (APUF) and its variants such as XOR-APUFs, is one of popular silicon PUF construction considering its compact structure and the large challenge-response pair (CRP) space characterizing of a strong PUF [8].

Prominent PUF applications include authentication and cryptographic key generation [1]. In recent years, use of PUFs in more advanced cryptographic protocols such as key exchange, oblivious transfer, bit commitment and multi-party computation [9], [10], where a strong PUF is always required, has been investigated. Although authentication is able to tolerate the noisy PUF responses, key generation applications and key exchange, especially recent advanced cryptographic protocols [9], require large number of error free responses.

Stabilizing PUF response is usually left to the on-chip error correcting logic assisted by the associated priori computed helper data as illustrated in Fig. 1. Such a scheme maybe adequate for a single or a small number of key extractions such as from SRAM PUFs [7], but becomes a limitation when a large number of keys are necessary. We propose exploiting PUF models, originally used attack PUFs by creating PUF functional copies [8], to enable the determination of a randomly selected challenge to generate error-free response under all operating conditions in the absence of the APUF. The developed approach is able to: i) eschew expensive error correction logic on PUF embedded security modules; ii) overcome the burden of exhaustive characterization of huge number of CRPs, especially when large number of keys extracted from strong PUFs are necessary; iii) eliminate computation of helper data and subsequent on-chip storage or off-chip storage and transfer. We summarize our contributions below:

- a) We exploit statistical delay models of APUFs to evaluate the response reliability of a random challenge to select challenges that generate error-free responses under all operating conditions.
- b) We have validated our approach based on two APUF datasets. First, we build synthetic APUFs using real-world RO-PUF [6] frequency measurements to acquire arbitrary number of CRPs for performing extensive evaluations. Second, we use 64,000 CRPs collected across eight APUFs implemented on eight FPGAs. Our results show that no erroneous bits occur when selected responses are re-evaluated across a wide range of operating conditions. In addition, the selected challenges still exhibit the randomness characteristic expected from the corresponding responses.
- c) We show that our error-free response generation method is 'cost-free' to the PUF integrated device without any area or power overhead. Building a model takes less than fifteen seconds including collecting a small number, 10,000, of CRPs and the subsequent delay-time statistical characterization of each delay segment at each stage of an APUF, while the reliable challenge selection using the

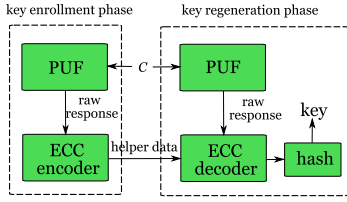


Fig. 1. Generalized ECC based PUF key generation scheme.

model can be performed on demand.

Next section introduces related work. Section III describes the approach to build an accurate statistical delay model of an APUF, and subsequently describes an algorithm to filter challenges that produce highly reliable response bits. Section IV validates the proposed approach based on two APUF datasets. Section V concludes this paper.

II. RELATED WORK

Reconciling PUF response errors is usually carried out by employing error correction codes (ECC). Thus the ECC based generalized PUF key generations have two phases as illustrated in Fig. 1: (1) The key enrollment phase computes helper data that will be used in (2) the key regeneration phase to reconcile errors of the reproduced response and retrieve the enrolled key.

This scheme is efficient to guarantee a small number of error-free responses to derive a single cryptographic key such as from a SRAM PUF [7], but it becomes expensive when multiple number of keys are necessary to realize: i) revocation of keys; ii) session key exchange; iii) controlled PUF constructions [4], [11]. These applications all prefer employing a strong PUF that is able to provide a large number of error-free responses. Notably, the associated helper data leaks information that may be exploited to attack PUFs [12].

There is a concurrent but independent investigation of selecting reliable responses of delay-based PUFs [13] using simulated data, where the effect of explicit PUF operational conditions, such as operating voltage and temperature, on the selection strategy remains to be investigated. Based on measured data, we extensively evaluate our error-free response generation approach across a wide range of operating conditions.

III. ERROR FREE RESPONSE SELECTION

We assume that a trusted party—the server—has one-time access to the underlying APUFs acting as a PUF building block within the controlled PUF or PUF embedded cryptographic key generators. The PUF building block can also be XOR-APUFs rather than basic APUFs. The server securely stores the derived statistical model of each APUF and destroys the one-time access, eg., fusing one-time programmable wires. The challenge response interface can be protected. For example, in a controlled PUF construction, it is achieved in a way of $g(x) = h_2(h_1(x), f(h_1(x)))$, where h is a hash function, x is the input, and $f(\cdot)$ is the PUF.

We employ a delay time adaptive characterization technique following [14] to build up a statistical model of an APUF

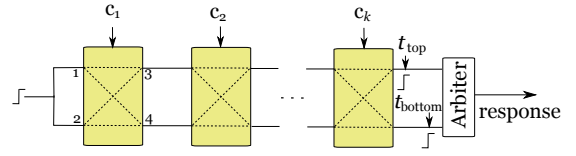


Fig. 2. Structure of an arbiter PUF (APUF). Signal propagation delays of the top path t_{top} and the bottom path t_{bottom} in the arbiter input determines the response bits.

that is able to accurately estimate the internal delay times of the APUF to eschew physically characterizations. In [14], the authors employ this adaptive characterization to realize an impersonation of an APUF from a physical means rather than a software means. During the key extraction phase, filtered reliable challenges are issued by the server and applied to the PUF to obtain reliable response bits that can be directly utilized to derive keys without error correction. Next, we detail the principles of generating error free responses when wire delay times for all stages of the APUF are estimated.

A. Arbiter PUF

As depicted in Fig. 2, an APUF consists of k stages of two 2-input multiplexers, or any other unit forming two theoretically symmetrical, but practically asymmetrical, signal paths due to inherent randomness in the fabrication processes. To generate a response bit, an active pulse is fed as an input to the first stage, while the selection bit of c_i , $i \in \{1, \dots, k\}$ determines the signal path to the next stage. For example, if $c_i = 1$, two signals propagate from 1 to 3 and from 2 to 4, respectively, in other words, these two signals pass through the i -th stage of the APUF without crossing. Conversely, if $c_i = 0$, they propagate from 1 to 4 and from 2 to 3 concurrently, these two paths are referred to as cross paths. We refer to the delay time from 1 to 3 of i -th stage as t_{13}^i , and the corresponding delays through the others as t_{14}^i, t_{23}^i , and t_{24}^i . At the circuit level, each stage of the APUF can be implemented by a multiplexer. At the end of the cascaded multiplexers, an arbiter, which can be implemented by a latch, determines whether the top or bottom signal arrives first and hence results in a logic '0' or '1', accordingly, to yield a 1-bit response.

B. Response Reliability

More specifically, the delay time difference t_{dif} between the delay time of the top path t_{top} and the delay time of the bottom path t_{bottom} determines the response bit (output) at the arbiter and also its reliability, as depicted in Fig. 3. For a randomly chosen challenge, if the corresponding delay time difference $t_{\text{dif}} > 0$, then a response of '0' is produced, and vice versa. Further, if $|t_{\text{dif}}| \leq |\Delta t|$, Δt being a delay discrimination threshold, the delay time difference t_{dif} falls into the unreliable region as illustrated in Fig. 3; then the corresponding response bit has a high probability of being nondeterministic during repeated regeneration attempts, especially under varying environmental conditions. Conversely, if $|t_{\text{dif}}| > |\Delta t|$, then a highly reliable response bit can be expected. Further, if $|\Delta t|$ is far away from zero, we can expect that a response bit without

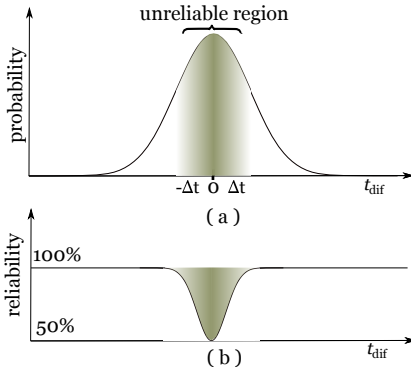


Fig. 3. (a) Delay time difference t_{dif} distribution, where $t_{\text{dif}} = t_{\text{top}} - t_{\text{bottom}}$, t_{top} and t_{bottom} are the signal propagation delay times of the input signal at the top path and the bottom path respectively. (b) Response bit reliability determined by t_{dif} to a given challenge.

error will be generated. This relies on the fact that responses to those challenges resulting in a large $|t_{\text{dif}}|$ are able to tolerate a greater degree of environmental variations, eg., fluctuation of supply voltage, and temperature, as further depicted in Fig. 4. For example, considering that the t_{top} and t_{bottom} have different simplified linear temperature co-efficients. A large t_{dif} guarantees no intersection between t_{top} and t_{bottom} , while an intersection eventually results in a flipped response—erroneous response when the temperature deviates. Similar observation has shown in [1], where a pair of ring oscillators (ROs) exhibiting a large frequency difference ensures a reliable response bits regenerated from a ROPUF.

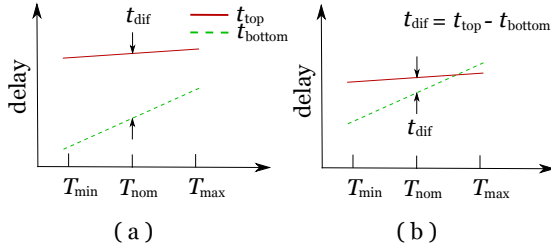


Fig. 4. Delay time as a function of the temperature. (a) Challenge always reproduces the response with 100% reliability within the temperature range from T_{min} to T_{max} . (b) Challenge results in an unreliable response bit.

If $t_{13}^i, t_{14}^i, t_{23}^i$, and t_{24}^i can be accurately measured, then it is easy to determine challenges capable of generating highly reliable response bits by evaluating $|t_{\text{dif}}|$. Concurrently, the response bit value, being ‘0’ or ‘1’, can also be judiciously determined. Physical measurement of each delay segment is hard, if not impossible, in practice. It may require probing the die which may damage the circuit or alter the delay, or other on-chip complicated peripheral circuits that are always unavailable for resource-constraint devices [15]. Next, we introduce a method following [14] for obtaining a statistical model to capture delay times of internal wires for all k stages of an APUF.

C. Modeling an APUF

The goal of a statistical model is to accurately characterize individual delay times of $t_{13}^i, t_{14}^i, t_{23}^i$, and t_{24}^i . More specifically, if the probability of $t_{13}^i > t_{24}^i$, $P(t_{13}^i > t_{24}^i)$, and probability of $t_{14}^i > t_{23}^i$, $P(t_{14}^i > t_{23}^i)$, can be estimated precisely, then $P(t_{13}^i > t_{24}^i)$ and $P(t_{14}^i > t_{23}^i)$ can be used to replace the delay time difference of $t_{13}^i - t_{24}^i$ and $t_{14}^i - t_{23}^i$ at the i -th stage of an APUF. In other words, we need to statistically estimate the following four probabilities in (1) after collecting a number of CRPs.

$$\begin{aligned} P_{13}^i &= P(t_{13}^i \text{ is longer than } t_{24}^i \text{ at } i\text{-th stage}) \\ P_{24}^i &= P(t_{24}^i \text{ is longer than } t_{13}^i \text{ at } i\text{-th stage}) \\ P_{14}^i &= P(t_{14}^i \text{ is longer than } t_{23}^i \text{ at } i\text{-th stage}) \\ P_{23}^i &= P(t_{23}^i \text{ is longer than } t_{14}^i \text{ at } i\text{-th stage}) \end{aligned} \quad (1)$$

Given the probabilistic formula of delays, there exists two relations $P_{13}^i + P_{24}^i = 1$ and $P_{14}^i + P_{23}^i = 1$ for $i \in \{1, 2, \dots, k\}$.

The $P_{13}^i, P_{24}^i, P_{14}^i, P_{23}^i$ for all stages are estimated based on an adaptive characterization technique. We refer readers to Xu *et al.*'s work for detailed implementations [14]. Consequently, a statistical delay model of an APUF is obtained when $P_{13}^i, P_{24}^i, P_{14}^i, P_{23}^i$ for all stages are estimated. As there is a linear relation between the delay time, eg., t_{12} , and its corresponding probability, eg., P_{12} [14], we can use the corresponding probability to replace the delay time and consequently to predict responses and concurrently their reliability for unseen challenges according to t_{dif} that is the summation of delay difference for all stages.

D. Determining Reliable Challenges

Filtering reliable response bits is straightforward and simple, once the statistical model is obtained, and it is a one-time task done during the enrollment phase by the server—the trusted authority. Most importantly, it costs no extra area and power overhead to the PUF integrated device, while it also avoids the overhead related to the ECC and helper data during both, key enrollment and regeneration phases. The challenge filtering procedure is described in the **Algorithm 1**.

Algorithm 1 Filtering challenges that generate reliable responses based on the statistical models of APUFs

```

1: procedure selection (a randomly chosen challenge  $C$ , PUF
   model  $f(\cdot)$ ,  $\Delta t$ )
2:    $t_{\text{dif}} \leftarrow f(C)$ 
3:   if  $t_{\text{dif}} < -\Delta t$  then
4:     response  $\leftarrow$  0; select  $C$ ;
5:     return
6:   else if  $t_{\text{dif}} > \Delta t$  then
7:     response  $\leftarrow$  1; select  $C$ ;
8:     return
9:   else
10:    discard  $C$ ;
11:  return

```

IV. EXPERIMENTAL VALIDATIONS

In this section, we empirically evaluate the reliable challenge selection method employing two datasets: i) we build up synthetic APUFs using ROPUFs [16] frequency measurements, by this means, we are able to obtain almost arbitrary number of CRPs for extensively testings; ii) we use the CRP data collected across eight APUFs implemented on eight FPGAs [17], the total number of CRPs in this dataset is 64,000, which is the main reason that we consider the synthesized APUFs first to obtain arbitrary number of CRPs due to 64,000 CRPs insufficient for some tests. For example, by using the first dataset, we test up to 50 million reliable challenges, all yielding error free responses, filtered from 830 million random challenges. This large number empirical tests is hard to be carried out based on the second dataset.

Building a statistical model of an APUF only requires collecting a number, 10,000, of CRPs under the nominal condition rather than all operating conditions. Time of CRP collection costs less than one second considering that one CRP evaluation needs 50 ns for a 64-stage APUF [5]. The delay characterization takes only less than fifteen seconds to obtain a model, where we use MATLAB 2012b to achieve the model building and the processor is an Intel i7-3770CPU@3.4GHz CPU.

A. Dataset Descriptions

There are five ROPUFs implemented across five Spartan3E S500 FPGAs boards in Virginia Tech's public ROPUF data. Each FPGA implements one ROPUF that consists of 512 ring oscillators (ROs). Detailed implementation information is detailed in [18]. The dataset contains each RO's frequency measurements. Each RO's frequency is measured 100 times under 0.96 V, 1.08 V, 1.20 V, 1.32 V, 1.44 V, respectively, at a fixed temperature of 25°C to reflect supply voltage influence. Similarly, each RO is also evaluated 100 times under 35°C, 45°C, 55°C, 65°C, respectively, with a fixed supply voltage of 1.20 V, to reflect influence from temperature changes. To use these measurements for synthesizing an APUF, we employ the inverse of frequencies of four ROs to replace $t_{13}^i, t_{24}^i, t_{14}^i, t_{23}^i$ in order to form the realistic delay time behavior of the i -th stage in the APUF. By this means, we are able to exploit existing real time delay data to form a 64-stage synthesized APUF by using 256 ROs implemented in the same FPGA board. The arbiter (performing delay time comparison) function is carried out off-chip by post-processing in MATLAB. Consequently, we obtain five synthesized APUFs.

The second CRP dataset are obtained from eight PDL (programmable delay line) APUFs; each has 128 stages. Each APUF is fed with 64000 challenges, therefore, 64000 CRPs are collected [19], [20]. For each CRP, it is evaluated 128 times the same operating condition. In total, nine operating conditions are considered: (5°C, 0.95 V); (5°C, 1.00 V); (5°C, 1.05 V); (35°C, 0.95 V); (35°C, 1.00 V); (35°C, 1.05 V); (65°C, 0.95 V); (65°C, 1.05 V); (65°C, 1.05 V). We treat (35°C, 1.00 V) as the nominal condition.

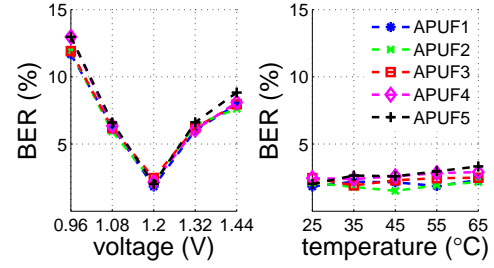


Fig. 5. Bit error rate (BER) under different supply voltage and temperature settings. Results are in well agreement with other experimentally reported results [21], [22].

B. Results of Synthesized APUFs

We evaluate each APUF's bit error rate (BER). The BER is the probability that two PUF responses from two distinct and random evaluations subject to the same randomly chosen challenge applied to the same PUF are different. In practice, a reference response took under the nominal condition is always used, while the regenerated response took under a different operating condition is compared with the reference response [21]. In our tests, each response bit is evaluated 100 times given the same challenge under the same operating condition. Results are shown in Fig. 5. The BER solely introduced by noise is around 2.2% when both voltage and temperature are under the nominal condition, where the voltage is 1.2 V and the temperature is 25°C. We can see that supply voltage has predominant influence on the APUF BER compared with the temperature. Further, the worst-case BER of 12.98% is observed when the voltage has a -20% deviation from the nominal condition. Although APUFs evaluated here are synthesized, the presented reliability performance is in good agreement with [21], [22].

In the following descriptions, for convenience, we refer to a challenge that satisfies $|t_{\text{dif}}| > \Delta t$ as a reliable challenge. Concurrently, we refer to the response bit corresponding to a reliable challenge as a reliable response. The statistical model has 97% response accuracy prediction that is well agreed with [14]. After reliable challenges are selected by performing the challenge filtering **Algorithm 1**, they are applied to the same APUF when operating conditions are changed to any setting within a wide range. The regenerated response bits are compared with the reference response bit evaluated at the nominal condition. If they are same, this indicates that such a filtered challenge does have a very high tolerance to the environmental changes, which guarantees an error free response bit regenerated across a wide operating conditions. Otherwise, an error is marked and counted. We calculate the error rate that is the percentage of counted errors out of the number of selected reliable challenges. It is convenient to refer the error rate under a specific Δt setting as $\text{BER}@\Delta t$. We denote the $\text{BER}@\Delta t$ as below.

BER@ Δt . The $\text{BER}@\Delta t$ is the probability that the reevaluated response bit took under one random chosen operating condition within a range is different from the reference re-

TABLE I
BER@ Δt . SYNTHETIC APUF OPERATIONAL RANGE: SUPPLY
VOLTAGE(1.08-1.32 V), TEMPERATURE(25°C-65°C)

APUF No	BER@ Default	BER@ (Δt = 0.5)	BER@ (Δt = 0.75)	BER@ (Δt = 1.0)	BER@ (Δt = 1.25)	BER@ (Δt = 1.5)
1	6.30%	0%	0%	0%	0%	0%
2	6.43%	0.013%	0%	0%	0%	0%
3	6.21%	0.00205%	0%	0%	0%	0%
4	6.34%	0.007%	0%	0%	0%	0%
5	6.60%	0.01%	0.0008%	0%	0%	0%

TABLE II
BER@ Δt . SYNTHETIC APUF OPERATIONAL RANGE: SUPPLY
VOLTAGE(0.96-1.44 V), TEMPERATURE(25°C-65°C)

APUF No	BER@ Default	BER@ (Δt = 0.5)	BER@ (Δt = 0.75)	BER@ (Δt = 1.0)	BER@ (Δt = 1.25)	BER@ (Δt = 1.5)
1	11.68%	0.91%	0.1338%	0.0041%	0%	0%
2	11.90%	0.18%	0.13%	0.0056%	0.0006%	0%
3	11.92%	1.44%	0.22%	0.0023%	0.001%	0%
4	12.97%	1.29%	0.18%	0.0115%	0%	0%
5	12.98%	1.76%	0.34%	0.05%	0.0008%	0%

sponse bit evaluated under the nominal operating condition by applying the same selected reliable challenge to the same PUF, where the reliable challenge resulted delay time difference between top and bottom paths in an APUF satisfying $|t_{\text{dif}}| > \Delta t$.

We also refer BER@($\Delta t = 0$) as BER@Default. In fact, BER@Default is the BER when challenge filtering is not employed, eg., shown in Fig. 5.

1) *Reliability of Filtered Responses*: Tested results are shown in Table. I and II. The BER@ Δt is evaluated within different operating ranges for each Table. Noting the supply voltage range of Table. II ($\pm 20\%$ deviation) is larger than Table. I ($\pm 10\%$ deviation). It is not surprising that BER@ Δt is not zero when the Δt is set to a small value. But as the Δt goes up, BER@ Δt decreases significantly. When the $\Delta t = 1.5$, all filtered challenges that satisfy $|t_{\text{dif}}| > (\Delta t = 1.5)$ produce responses with 100% reliability. The APUF₅ is the most interested testing sample because, in Fig. 5, APUF₅ has the highest worst-case BER of 12.98% among other APUFs when the voltage varies between 0.96 V and 1.44 V. By using judiciously selected error free responses to derive keys, it equals to an ECC decoder that has to correct at least 12.98% errors.

Due to infinite number of reliable challenges filtering cannot be achieved in evaluation, we use up to more than 830 million randomly generated challenges for all testings, among them, 50 million selected reliable challenges are obtained when the $\Delta t = 1.5$. We can see that all of responses given selected challenges are error free regenerated even when the supply voltage has a $\pm 20\%$ variation and the temperature experiences a 40°C variation.

2) *CRP Loss*: The CRP loss is the probability that a randomly given challenge cannot meet the challenge filtering criterion of $|t_{\text{dif}}| > \Delta t$. The CRP loss is increasing as the Δt increases. Fig. 6 depicts the CRP loss as a function of Δt . We can see that 94% response bits will be discarded

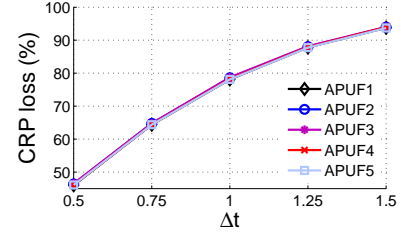


Fig. 6. Illustration of CRP loss induced by the reliable challenge filtering.

TABLE III
BER@ Δt . PDL APUF OPERATIONAL RANGE: SUPPLY
VOLTAGE(0.95-1.05 V), TEMPERATURE(5°C-65°C)

APUF No	BER@ Default	BER@ (Δt = 0.5)	BER@ (Δt = 1.0)	BER@ (Δt = 1.5)	BER@ (Δt = 1.6)	BER@ (Δt = 1.7)
1	10.57%	1.26%	0.095%	0%	0%	0%
2	16.68%	8.05%	1.46%	0.039%	0%	0%
3	8.59%	1.08%	0.037%	0%	0%	0%
4	8.62%	0.656%	0.013%	0%	0%	0%
5	11.52%	0.08%	0%	0%	0%	0%
6	11.07%	0.71%	0.031%	0%	0%	0%
7	5.87%	1.35%	0.032%	0%	0%	0%
8	15.00%	5.98%	0.89%	0.056%	0.085%	0%

when $|t_{\text{dif}}| > (\Delta t = 1.5)$. As an APUF is able to generate an exponential number of CRPs as a function of the number of stages k , large number of reliable challenges can still be promised. For example, given a 64-stage APUF, its CRP space is up to 2^{64} . Hence, there are still $2^{64} \times 0.06 > 2^{59}$ error free response bits available. When the PUF operating range varies insignificant in practice, a smaller Δt can be chosen to decrease the CRP loss while still produce error free responses as shown in Table. I.

C. PDL APUFs

The statistical model is obtained by training 10,000 CRPs evaluated only under the nominal condition and the prediction accuracy with 92.41% is achieved. This lower prediction accuracy compared with the prediction accuracy of the synthesized APUF model are attributed to [9]: i) The slightly increased nominal BER is 4.99% that is obtained under the nominal condition—the nominal BER of the synthesized APUF is 2.2%, see Fig. 5; ii) The usage of PDLs on FPGA, which makes the MUX structure more complicated (slightly nonlinearity is introduced).

There are eight PDL APUFs tested, each of them is implemented on a different FPGA board. The maximum BER@ Δt for all of eight APUFs are listed in Table. III. Among tested eight APUFs, the maximum worst-case BER@Default is 16.68%. When the $\Delta t = 1.7$, there is no error found in all reproduced responses under any tested operating condition. Same to the results obtained from synthesized APUFs in II, BER@ Δt decreases as the Δt increases. There are still around 1% challenges satisfying the selection criterion for all eight tested APUFs even when $\Delta t = 1.7$.

We further investigate the response randomness—percentage of occurrence of ‘1’s in response bits—relationship

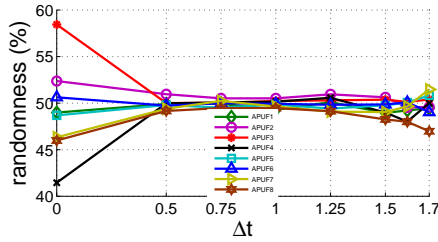


Fig. 7. Randomness under different Δt settings.

with the Δt . In general, the challenge filtering should not result into a bias to PUF's response with a preferable value ('0'/'1'). In Fig. 7, the challenge filtering does not deteriorate the randomness of the selected reliable response bits. The randomness is always close to 50% when the challenge filtering is performed under different Δt settings. In addition, even the response's randomness is not close to 50% initially, the challenge filtering seems eliminate such an initial randomness bias.

V. CONCLUSION

We propose an approach for determining error free responses even when they are re-evaluated across a wide range of operating condition. The proposed approach can be adopted into controlled PUF designs and also provide large number of error free responses on demand for advanced cryptographic applications. This method has no burden to the PUF embedded devices because there is no ECC and helper data involved, where the model building is left to the server and only asks negligible computational resources to the server. Extensive empirically evaluations validate the practicability of our error free response methodology. There are some interesting future works. Firstly, though good results have been demonstrated through simulated data [13] considering aging effects, further empirical validations remain to be investigated. Secondly, adopting our error-free response generation method in to controlled PUFs [11]. In this context, it is imperative to reduce the CRP loss, which will enable an efficient control logic realization within the controlled PUFs. Last but not least, attacks assisted by the helper data compromising the security of a key generator only extracting a single key or very limited number of keys [12] may be even harder, if not impossible, when they are mounted to key generators in capable of generating a large number of keys where the helper data is not used. This will be another interesting investigation of securely using error-free responses [23].

VI. ACKNOWLEDGMENT

We acknowledge the APUF dataset provided by Dr Mehrdad Majzoobi and Mr Siam U. Hussain from research group of Prof Farinaz Koushanfar.

REFERENCES

[1] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proceedings of the 44th Annual Design Automation Conference*. ACM, 2007, pp. 9–14.

[2] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal, "Design and implementation of PUF-based" unclonable" RFID ICs for anti-counterfeiting and security applications," in *IEEE International Conference on RFID*. IEEE, 2008, pp. 58–64.

[3] S. U. Hussain, M. Majzoobi, and F. Koushanfar, "A built-in-self-test scheme for online evaluation of physical unclonable functions and true random number generators," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 2, no. 1, pp. 2–16, 2016.

[4] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Controlled physical random functions," in *18th Annual Computer Security Applications Conference*. IEEE, 2002, pp. 149–160.

[5] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 10, pp. 1200–1205, 2005.

[6] A. Maiti, I. Kim, and P. Schaumont, "A robust physical unclonable function with enhanced challenge-response set," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 1, pp. 333–345, 2012.

[7] Holcomb, Daniel E, W. P. Burleson, and K. Fu, "Power-up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1198–1210, 2009.

[8] U. Ruhrmair, J. Solter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas, "PUF modeling attacks on simulated and silicon data," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1876–1891, 2013.

[9] U. Ruhrmair and M. Van Dijk, "PUFs in security protocols: Attack models and security evaluations," in *IEEE Symposium on Security and Privacy (SP)*, 2013, pp. 286–300.

[10] M. van Dijk and U. Ruhrmair, "Protocol attacks on advanced PUF protocols and countermeasures," in *Proceedings of the conference on Design, Automation & Test in Europe*. European Design and Automation Association, 2014, p. 351.

[11] B. Gassend, M. V. Dijk, D. Clarke, E. Torlak, S. Devadas, and P. Tuyls, "Controlled physical random functions and applications," *ACM Transactions on Information and System Security*, vol. 10, no. 4, p. 3, 2008.

[12] G. T. Becker, "On the pitfalls of using arbiter-PUFs as building blocks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 8, pp. 1295–1307, 2015.

[13] X. Xu, W. Burleson, and D. E. Holcomb, "Using statistical models to improve the reliability of delay-based PUFs," in *IEEE Computer Society Annual Symposium on VLSI*, 2016, DOI: 10.1109/ISVLSI.2016.125.

[14] T. Xu, D. Li, and M. Potkonjak, "Adaptive characterization and emulation of delay-based physical unclonable functions using statistical models," in *Proceedings of the 52nd Annual Design Automation Conference*. ACM, 2015, p. 76.

[15] M. Majzoobi, E. Dyer, A. Elnably, and F. Koushanfar, "Rapid FPGA delay characterization using clock synthesis and sparse sampling," in *International Test Conference (ITC)*, 2010, DOI: 10.1109/TEST.2010.5699248.

[16] <http://rijndael.ece.vt.edu/variability/main.html>.

[17] M. Majzoobi, A. Kharaya, F. Koushanfar, and S. Devadas, "Automated design, implementation, and evaluation of arbiter-based PUF on FPGA using programmable delay lines," *IACR Cryptology ePrint Archive*, vol. 2014, p. 639, 2014.

[18] A. Maiti, J. Casarona, L. McHale, and P. Schaumont, "A large scale characterization of RO-PUF," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2010, pp. 94–99.

[19] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Techniques for design and implementation of secure reconfigurable PUFs," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 2, no. 1, p. 5, 2009.

[20] M. Majzoobi, F. Koushanfar, and S. Devadas, "Fpga puf using programmable delay lines," in *IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2010, DOI: 10.1109/WIFS.2010.5711471.

[21] M. Roel, "Physically unclonable functions: Constructions, properties and applications," Ph.D. dissertation, University of KU Leuven, 2012.

[22] R. Maes, V. Rožić, I. Verbauwhe, P. Koeberl, E. Van der Sluis, and V. Van der Leest, "Experimental evaluation of physically unclonable functions in 65 nm CMOS," in *Proceedings of the ESSCIRC*. IEEE, 2012, pp. 486–489.

[23] Y. Gao and D. C. Ranasinghe, "PUF-FSM: A controlled strong PUF," *arXiv preprint arXiv:1701.04137v2*, 2017.